DeepGeo: Photo Localization with Deep Neural Network (#10)

Montiel Abello | Nathaniel Chodosh | Sudharshan Suresh

Introduction and Motivation

- Games such as GeoGuessr show that humans are able to use visual cues such as landscape, vegetation, man-made infrastructure to roughly infer location from a single image. In this project we attempt to replicate this semantic reasoning using a deep neural network.
- DeepGeo will aid in applications such as meta-data aggregation, location-based priming and the creation of richer datasets.
- Previous works, based on both learning as well as scene matching methods, have shown appreciable results at various spatial scales (*IM2GPS* [1] and PlaNet [3]).
- *DeepGeo* targets the classification problem. We restrict the scope to classifying images into states in the United States. Near-perfect street view coverage, high road density and varied geography make the United States an ideal playground for this task.
- **Problem Definition:**
  Given a set of input samples $X = \{X_1, X_2, \dots, X_n\}$ where $X_i \in R^{4 \times 3 \times 256 \times 256}$, and a set of one-hot vector labels $Y = \{Y_1, Y_2, \dots, Y_n\}$ where $Y_i \in R^{50}$, we create a model to predict the distribution over all states of a sample belonging to a given class, $P(Y' = Y_i | X')$.

Key Contributions:

a) **DeepGeo**: a deep neural network classifier that predicts the most-probable state a given image was captured in (not restricted to cities/landmarks).
b) The **50State10K** dataset, the first comprehensive street view dataset of the States in the USA.

Methods : Dataset and Sampling

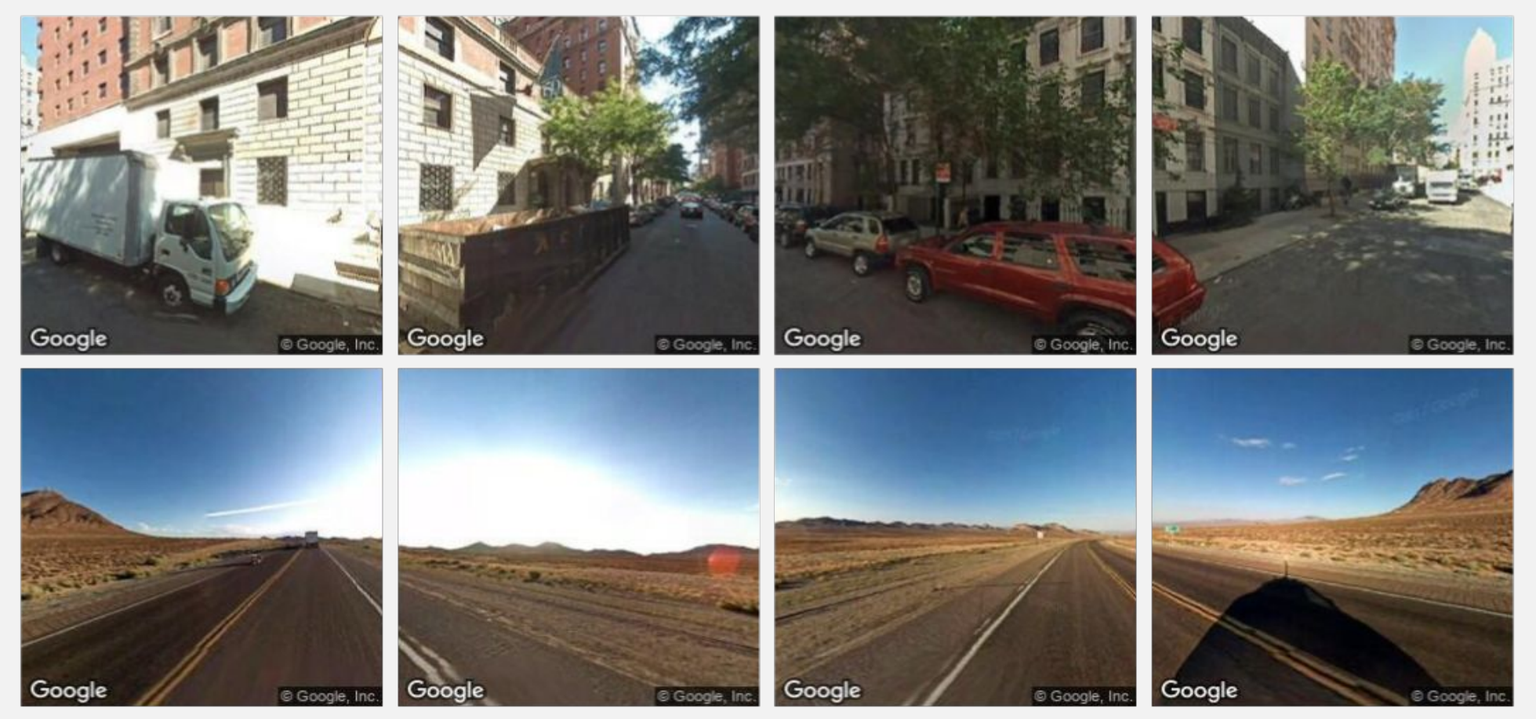

Figure 1: Two input samples consisting of four images each at headings.

- Our dataset `50States10K` consists of Google Street View images, via the Google Street View API.
- For a given sample location, the closest available Street View panorama is extracted and split into 4 rectified images facing the cardinal directions.
- To sample locations within each state, state boundary data was used. Population density was used to inform location sampling, improving the hit-rate of the panorama extractor and providing richer data.
- A population density map with grid size of 2.5 arc-minutes in latitude and longitude was mapped to the state boundary polygon set.
- For training purposes, 2500 unique locations were sampled in each of the 50 states to provide a set of $4 \times 2500 \times 50$ unique images.
- For testing, 500 locations in each state were sampled to produce a set of $4 \times 500 \times 50$ images.

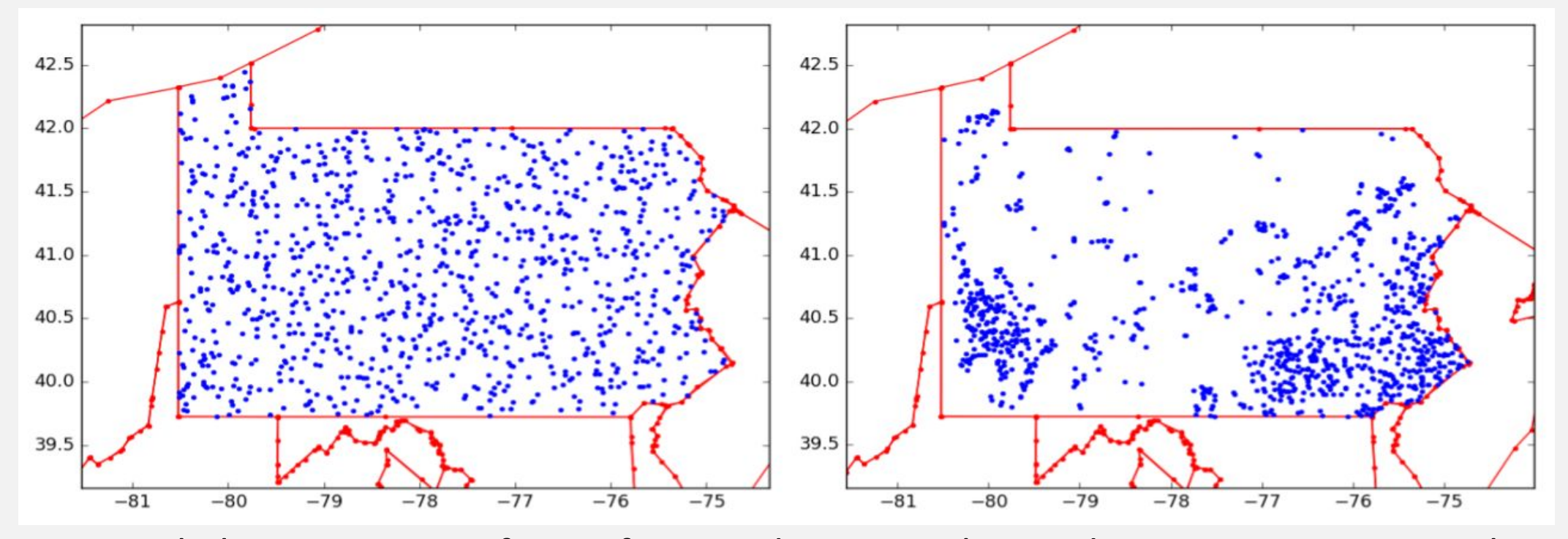

Figure 2. Sample locations in PA for uniform and our population density aware approaches.

Methods : Network and Training

- ResNet is a recently published CNN architecture which we determined to be suitable for our problem [2]. It uses so called *residual blocks* to achieve better gradient propagation, which enables training much deeper networks. We determined empirically that the 50-layer variant was sufficient.
- As a baseline method we performed standard CNN classification, treating each image as a separate training example.
- We then tested **3 variants** of this network to determine the best way of integrating the information from multiple views of the same location. Some previous works [3] have used sophisticated LSTM setups to accomplish this, but for this case each input would have a fixed number of input images. Additionally the 4 input images have a fixed relationship, since they always correspond to views in cardinal directions. With this in mind we chose to use simpler integration techniques.

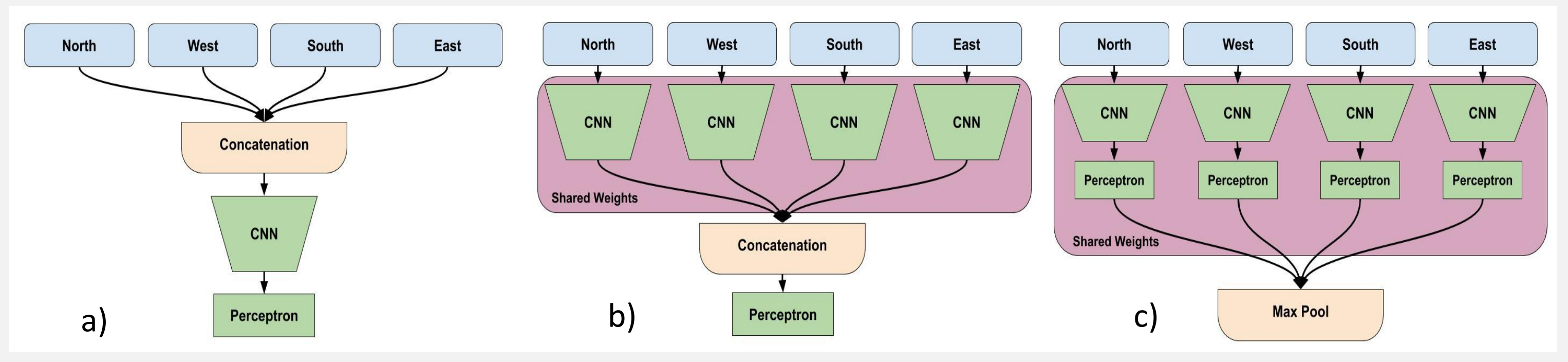

Figure 3. a) Early, b) Medium and c) Late integration network architectures.

- **Early Integration:** All 4 views are concatenated to form a 12 channel input image, allowing information to be shared between images in all layers of the network. The network is otherwise a standard 50-layer ResNet, the only modification being that the number of filters in each block of the network is doubled to account for the extra input channels.
- **Medium Integration:** We assume that the feature extraction for each of the input images should be constant, and that integration should be done only on the extracted features. To this end, we use a CNN to get a low dimensional summary of each image, concatenate the features of each, and run a single layer perceptron on the new feature for classification. The CNN is shared across all four images, and is simply the 50-layer resnet with the fully connected layer removed.
- **Late Integration:** We run a shared CNN over each input image and then combine the results. However, in this scheme the CNN includes the fully connected layer and to integrate the predictions we simply take the maximum over each output class. The idea being that the only information to be gained by integration is being able to use the *best view* for the predictions.
- We implemented the network in Tensorflow as described in [2] and used the Adam algorithm for training with a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_1 = 0.999$ and $\varepsilon = 10^{-8}$. We also applied weight decay with a weight of 0.0001. All three variants were trained for fifty epochs using a 90/10 train/validation split. The epoch which achieved the highest validation accuracy was then used on the test set. Training took approximately 30 hours per network running on a single Nvidia Titan X GPU.

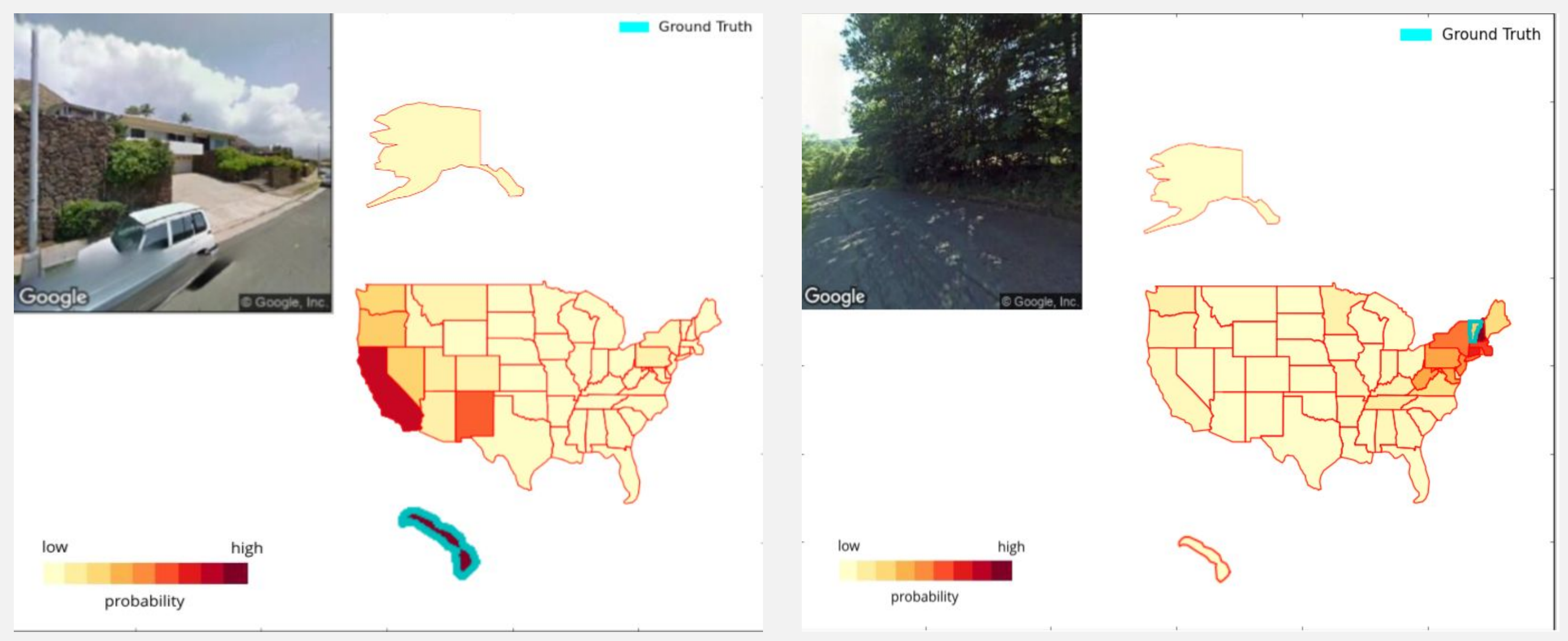

Figure 4. Heatmaps of (a) Hawaii (b) Vermont. These reflect the *distinctness* of features in states.

Results

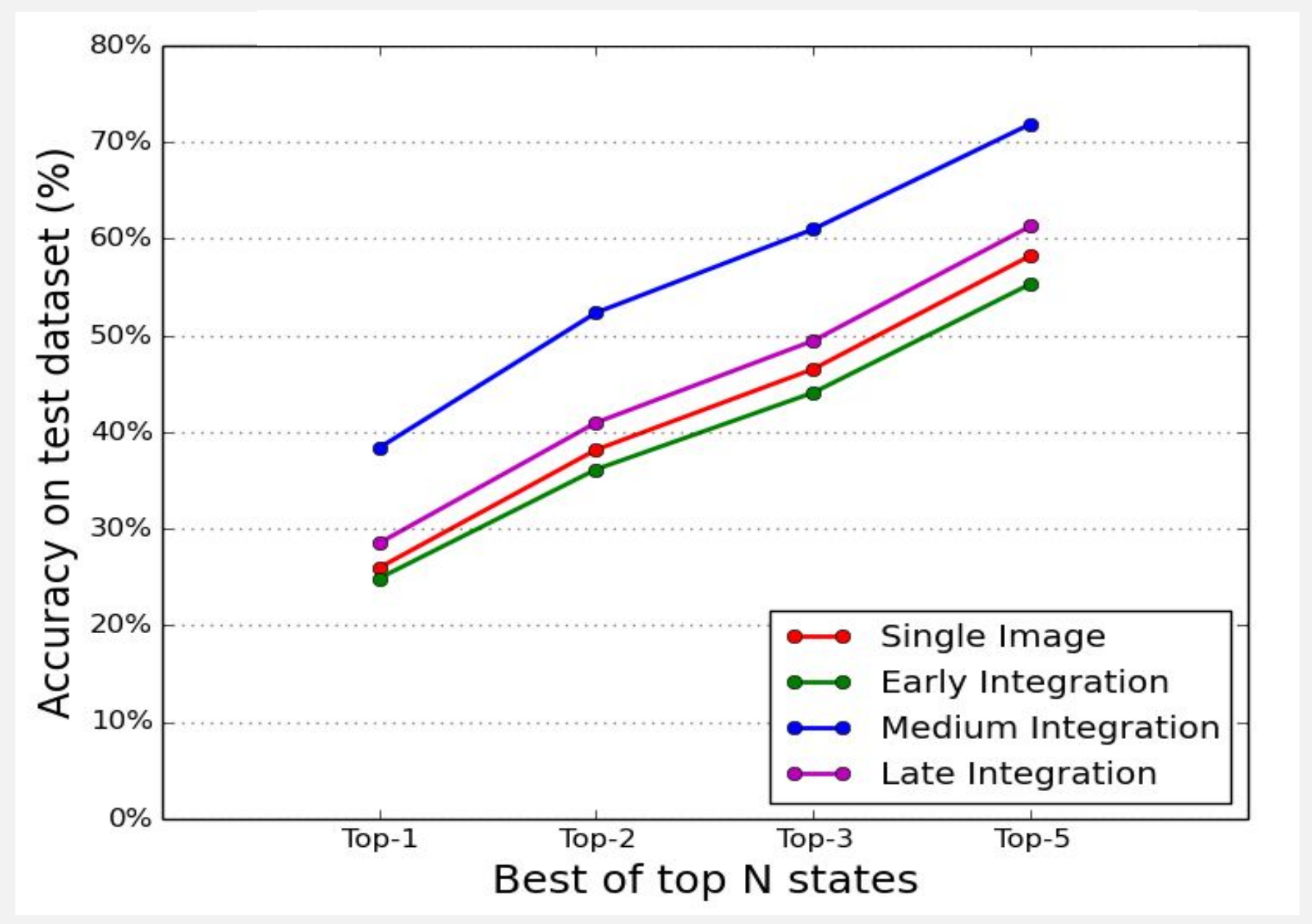

Figure 5. Best in top N accuracy for different network architectures

- **Dataset testing:** Accuracy results for the test dataset `50States2K` amongst the 4 different network architectures are shown above. We see that even in the single image case, a baseline accuracy of 25.92% is obtained. These improve significantly when considering the best in top-N results. The best performing network is the Medium integration one with 38.32%, which we shall use for all the subsequent results. Bear in mind that PlaNet correctly classifies 28.4% of the test data at the country level. Random chance for prediction is 2%.

Top 5		Bottom 5	
State	**Accuracy**	**State**	**Accuracy**
Hawaii	**0.956**	Virginia	**0.092**
Alaska	0.912	Texas	0.104
Arizona	0.734	Alabama	0.126
Wyoming	0.638	Ohio	0.144
Vermont	0.624	Missouri	0.168

Table 1. Best and worst class accuracies. Note that Hawaii and Alaska top the charts due to their characteristic appearances.

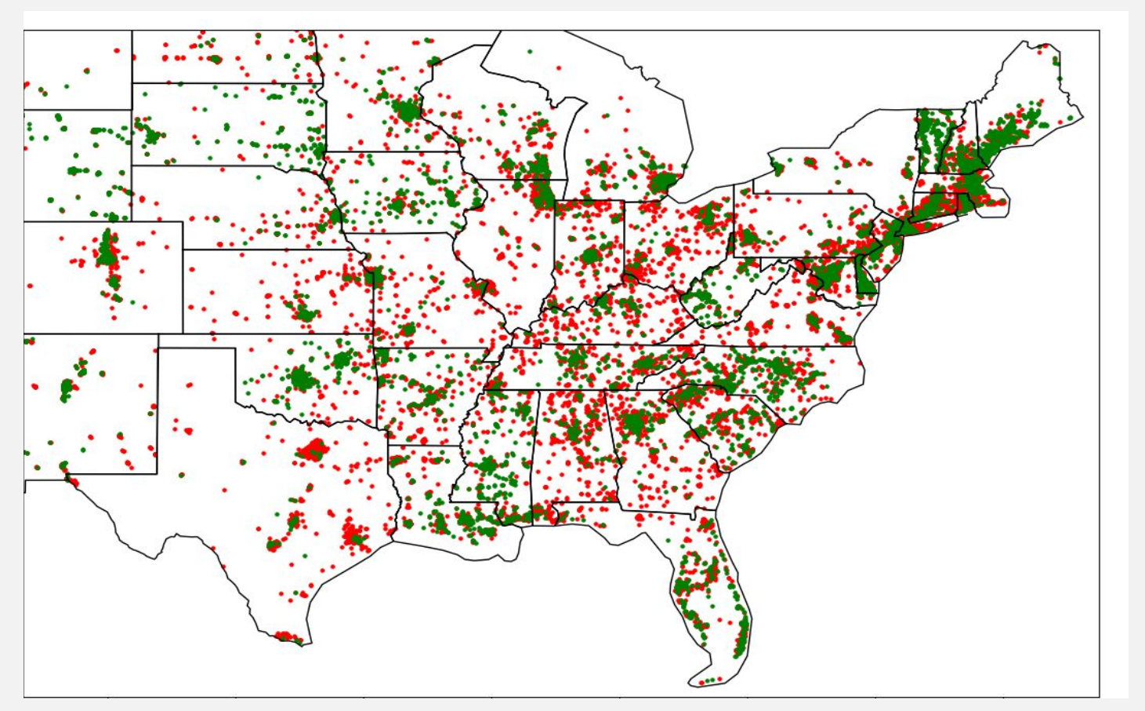

Figure 6. Test locations depicting correctly/incorrectly classified show that network performs poorly on samples from regions of low population density.

- **Human subject testing:** The performance of our network was evaluated in head-to-head rounds of GeoGuessr against human subjects. 10 locations tests were performed per game. To ensure a fair test, humans were only allowed to rotate the panorama viewpoint and could not explore the map. At each location, coordinates were recorded and used to download images facing the cardinal directions for network testing. It was necessary to replay roughly 50% of the tests (with a different human player) as some locations were rejected by the Google Streetview API image downloader. Table 2 shows the results of these tests. The medium integration architecture beat humans in 4 out of 5 rounds.

Game	Human Accuracy	Medium Integration Accuracy
1	0.0	0.6
2	0.0	0.1
3	0.3	0.1
4	0.1	0.3
5	0.1	0.2

Table 2. Network performance against humans in games of Geoguessr

Conclusion and Future Work

The GeoGuessr game poses a challenging image classification task, since for many inputs it is impossible to fully determine where the photo was taken. We have presented several DeepGeo network architectures for this task, achieving excellent performance with the best variant. In addition, we open-sourced a dataset for similar tasks. We explored importance of choosing the appropriate information integration strategy for a classification task. Specifically, we have shown that in contrast to conventional wisdom, the earliest integration scheme is not always superior. Future work includes labelling training data with image coordinates in order to solve the regression problem, expanding the capabilities and usefulness of this tool.

References

[1] Hays, James, and Alexei A. Efros. "IM2GPS: estimating geographic information from a single image." *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008.
[2] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
[3] Weyand, Tobias, Ilya Kostrikov, and James Philbin. "Planet-photo geolocation with convolutional neural networks." *European Conference on Computer Vision*. Springer, Cham, 2016.

www.github.com/suddhu/DeepGeo
https://www.dropbox.com/s/wxheq6wb9bn8myf/50States10K.zip?dl=0